L'Attaque de Coppersmith sur le Bourrage Court (Short Pad Attack)

Adya Sarr Dylan Mona

Octobre 2025

Table des matières

1 Introduction et Contexte de l'Attaque

L'attaque à bourrage court (*Short Pad Attack*) [C96] cible le **chiffrement RSA** utilisant un **petit exposant** public e (souvent $e = 3$) lorsque le **même message** est chiffré plusieurs fois avec des bourrages aléatoires différents.

- **Système Cible:** RSA avec clé publique $\langle N, e \rangle$, où e est *petit* (Exemple: $e = 3$).
- **Scenario:** Un message M est chiffré deux fois avec des bourrages aléatoires distincts, r_1 et r_2.
- **Messages Chiffrés (Plaintext):** Le message original M est précédé d'un décalage binaire (shifting) et d'un remplissage.

$$M_1 = 2^m M + r_1$$
$$M_2 = 2^m M + r_2$$

  où M est le message réel, m est la longueur du bourrage (en bits), et r_1, r_2 sont les bourrages aléatoires.
- **Chiffrés (Ciphertext) Interceptés:**

$$C_1 \equiv M_1^e \pmod{N}$$
$$C_2 \equiv M_2^e \pmod{N}$$

- **Hypothèse Cruciale:** Les longueurs des bourrages r_1 et r_2 sont **petites**. Nous cherchons à quantifier la taille X de la différence $\Delta = r_2 - r_1$.

---

2 Outils Mathématiques : L'Algorithme LLL

Soit $L \subset \mathbb{Z}^n$ un réseau engendré par une base $B = \{b_1, \ldots, b_n\}$.

Théorème 1 (Borne de LLL, [FR95]). *L'algorithme LLL produit une base réduite $(b'_1, \ldots, b'_n)$ pour le réseau L en un temps polynomial en n et $\log(\|B\|_{\max})$, telle que le premier vecteur $b'_1 \in L$ satisfasse*

$$\|b'_1\| \leq 2^{\frac{n-1}{4}} (\det(L))^{1/n}.$$

---

3 Outils Mathématiques : Lemme de Howgrave-Graham

Ce lemme établit une condition suffisante sur la norme euclidienne pondérée d'un polynôme $g(x)$ pour garantir que si x_0 est une racine de $g(x)$ modulo une puissance de N, elle est aussi une racine de $g(x)$ sur $\mathbb{Z}$.

Lemme 1 (Howgrave-Graham, [HG97]). *Soit $g(x) = \sum_{i=0}^{n} c_i x^i$ un **polynôme univarié à coefficients entiers**. Soient X et m deux **entiers positifs** tels que :*

1. $g(x_0) \equiv 0 \pmod{N^m}$ *où* $|x_0| \leq X$.
2. *La norme euclidienne pondérée de $g(xX)$ est bornée par :*

$$\|g(xX)\|_2 = \sqrt{\sum_{i=0}^{n} (c_i X^i)^2} < \frac{N^m}{\sqrt{n}}.$$

*Alors $\mathbf{x_0}$ est une **racine** sur les **entiers**, c'est-à-dire $\mathbf{g(x_0) = 0}$.*

Démonstration simplifiée

On cherche à borner $|g(x_0)|$:

$$|g(x_0)| = \left| \sum_{i=0}^{n} c_i x_0^i \right|$$

En utilisant le fait que $|x_0| \leq X$, on majore :

$$|g(x_0)| \leq \sum_{i=0}^{n} |c_i||x_0|^i \leq \sum_{i=0}^{n} |c_i| X^i \cdot 1$$

On applique l'inégalité de Cauchy-Schwarz pour les vecteurs $(|c_0|X^0, \ldots, |c_n|X^n)$ et $(1, \ldots, 1)$:

$$\sum_{i=0}^{n} |c_i| X^i \leq \sqrt{\sum_{i=0}^{n} (|c_i|X^i)^2} \cdot \sqrt{\sum_{i=0}^{n} 1^2} = \|g(xX)\|_2 \cdot \sqrt{n}$$

En appliquant la condition 2. (où $n+1$ est le nombre de monômes) :

$$|g(x_0)| < \sqrt{n} \cdot \frac{N^m}{\sqrt{n}} = N^m$$

Puisque $g(x_0) \equiv 0 \pmod{N^m}$ (condition 1), $g(x_0)$ est un multiple de N^m. Comme $|g(x_0)| < N^m$, le seul multiple possible est 0, d'où $g(x_0) = 0$.

---

4 Théorème Général de Coppersmith

Théorème 2 (Racines Univariées Modulaires [C96]). *Soit*

$$N \in \mathbf{Z}, \quad f(x) \in \mathbf{Z}[x]$$

un polynôme unitaire de degré d. Posons

$$X = N^{(1/d)}.$$

Alors, il existe un algorithme qui, étant donnée la paire (N, f), trouve tous les entiers

$$x_0 \in \mathbf{Z} \quad \textit{tels que} \quad |x_0| < X \quad \textit{et} \quad f(x_0) \equiv 0 \pmod{N}.$$

Le temps d'exécution de cet algorithme est polynomial en $\log N$ *et est dominé par le coût de l'exécution de l'algorithme LLL.*

Principe de la Preuve

On veut montrer que le problème de trouver une petite racine modulo N peut se reformuler comme la recherche d'un polynôme entier à petits coefficients partageant cette racine. Soient :

- $N \in \mathbf{Z}$,
- $f(x) \in \mathbf{Z}[x]$ un polynôme unitaire de degré d,
- $x_0 \in \mathbf{Z}$ une petite racine telle que

$$f(x_0) \equiv 0 \pmod{N}.$$

On pose

$$X = N^{\frac{1}{d}},$$

ce qui définit la borne supérieure sur la taille de la racine que l'on recherche : $|x_0| < X$. Pour exploiter l'algorithme LLL, on construit un réseau dont les vecteurs correspondent à des polynômes qui s'annulent en x_0 modulo des puissances de N. Remarquons que si $f(x_0) \equiv 0 \pmod N$, alors pour tout entier $k \geq 1$:

$$f(x_0)^k \equiv 0 \pmod{N^k}.$$

On choisit un entier $m \approx \frac{\log N}{d \log X}$ afin d'obtenir un réseau suffisamment grand pour LLL. On définit alors les polynômes

$$g_{u,v}(x) = N^{m-v} x^u f(x)^v, \quad 0 \leq u \leq d-1,\ 0 \leq v \leq m.$$

Ces polynômes satisfont

$$g_{u,v}(x_0) \equiv 0 \pmod{N^m},$$

et peuvent donc servir de vecteurs pour construire le réseau. Comme x_0 peut être proche de X, qui peut être assez grand, on applique une mise à l'échelle des polynômes pour que les vecteurs du réseau aient des coefficients comparables :

$$g_{u,v}(x) \mapsto g_{u,v}(xX).$$

Sans cette mise à l'échelle, LLL pourrait donner un polynôme « court » pour la norme des coefficients, mais dont la valeur en x_0 est énorme, donc inutile pour retrouver la racine.

Cette étape est donc cruciale : elle permet à l'algorithme LLL de produire un polynôme $h(x)$ à petits coefficients qui partage la racine x_0 :

$$h(x_0) = 0 \quad \text{dans } \mathbf{Z}.$$

Notons w la dimension du réseau :

$$w = (m+1) \cdot d.$$

Matrice du Réseau (Illustration)

La matrice associée au réseau peut s'écrire, par exemple pour $d = 2$ et $m = 3$ (où $f(x) = x^2 + a_1 x + a_0$) comme suit:

$$\mathbf{B} = \left[\begin{array}{c|cccccccc} & x^0 & x^1 & x^2 & x^3 & x^4 & x^5 & x^6 & x^7 \\ \hline g_{0,0}(xX) & N^3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ g_{1,0}(xX) & * & XN^3 & 0 & 0 & 0 & 0 & 0 & 0 \\ g_{0,1}(xX) & * & * & X^2N^2 & 0 & 0 & 0 & 0 & 0 \\ g_{1,1}(xX) & * & * & * & X^3N^2 & 0 & 0 & 0 & 0 \\ g_{0,2}(xX) & * & * & * & * & X^4N & 0 & 0 & 0 \\ g_{1,2}(xX) & * & * & * & * & * & X^5N & 0 & 0 \\ g_{0,3}(xX) & * & * & * & * & * & * & X^6 & 0 \\ g_{1,3}(xX) & * & * & * & * & * & * & * & X^7 \end{array}\right]$$

D'après la borne de Hermite, pour un réseau L de dimension w, il existe un vecteur non nul $v \in L$ dont la norme satisfait :

$$\|v\| \leq \sqrt{w}\ \det(L)^{1/w}.$$

En choisissant m suffisamment grand, on garantit que

$$\|v\| \leq \frac{N^m}{\sqrt{w}}.$$

En appliquant l'algorithme LLL au réseau, on obtient un polynôme $h(x)$ correspondant à un vecteur relativement court. Grâce à la sortie de LLL et à la borne de Hermite, on a :

$$\|h(x_0 X)\| \leq \frac{N^m}{\sqrt{w}}.$$

Et puisque $h(x)$ est une combinaison linéaire des $g_{u,v}(x)$

$$h(x_0) \equiv 0 \pmod{N}.$$

D'après le lemme de Graham, cela implique que x_0 est une racine entière de $h(x)$:

$$h(x_0) = 0.$$

On a donc réussi à construire un polynôme $h(x)$ à petits coefficients partageant la racine x_0. Pour obtenir x_0, il suffit de calculer les racines entières de $h(x)$. (par exemple la recherche exhaustive sur intervalle borné [-X,X])

5 Application Détaillée à RSA avec $e = 3$

L'attaquant intercepte deux chiffrements C_1 et C_2 du même message M avec deux bourrages r_1 et r_2. On pose $\Delta = r_2 - r_1$, ce qui donne la relation affine $M_2 = M_1 + \Delta$.

5.1 Construction du Polynôme Modulaire

L'objectif est d'éliminer la variable du message M_1 des deux congruences :

1. $g_1(x) = x^3 - C_1 \equiv 0 \pmod{N}$ (où $x = M_1$)
2. $g_2(x) = (x + \Delta)^3 - C_2 \equiv 0 \pmod{N}$ (où $x = M_1$)

Le résultant $h(y) = \text{Res}_x(g_1(x), g_2(x))$ est un polynôme en Δ (ici $y = \Delta$) tel que $h(\Delta) \equiv 0 \pmod{N}$. Pour $e = 3$, le polynôme résultant a un degré $k = e^2 = 9$:

$$h(\Delta) = \Delta^9 + (3C_1 - 3C_2)\Delta^6 + (3C_1^2 + 21C_1C_2 + 3C_2^2)\Delta^3 + (C_1 - C_2)^3 \equiv 0 \pmod{N}$$

5.2 Application du Théorème de Coppersmith

Nous avons ramené le problème à la recherche d'une racine Δ pour le polynôme univarié $h(y) \equiv 0 \pmod{N}$, où :

- Le polynôme est $h(y)$ de degré $k = 9$.
- La racine inconnue est $\Delta = r_2 - r_1$, bornée par $|\Delta| < 2^m$.

Le Théorème de Coppersmith garantit que la racine Δ est trouvable efficacement si elle est inférieure à $N^{1/k}$, soit :

$$|\Delta| < N^{1/9}$$

5.3 Conclusion et Décryptage

La différence de bourrage Δ est récupérable si la longueur du remplissage m (où $|\Delta| < 2^m$) est inférieure à $\log_2(N^{1/9})$, c'est-à-dire moins de $1/9$ de la longueur de N.

Une fois Δ trouvée, on dispose d'une relation affine connue entre M_1 et M_2 : $M_2 = M_1 + \Delta$. L'attaquant peut alors utiliser le **Théorème de Franklin-Reiter** pour récupérer les messages chiffrés.

Théorème 3 (Franklin-Reiter, [FR95]). *Soit (e, N) une clé RSA publique. Soient M_1 et M_2 deux messages tels que $M_2 = f(M_1) \pmod{N}$, où $f(x) = ax + b$ est une application affine connue avec $b \neq 0$. Alors, connaissant (N, e, C_1, C_2, f), on peut retrouver M_1 et M_2 en temps polynomial en $\log N$.*

5.4 Généralisation

L'attaque est applicable à tout exposant public e :

- Le polynôme résultant a un degré $k \leq e^2$.
- La condition de succès est que la longueur du bourrage m soit inférieure à $\log_2(N^{1/e^2})$, c'est-à-dire environ $1/e^2$ de la longueur de N.
- Pour l'exposant recommandé $e = 2^{16} + 1 = 65537$, la borne est de $1/e^2 \approx 2.3 \times 10^{-10}$, rendant l'attaque totalement impraticable car la taille de bourrage permise est quasiment nulle.

Références

Références

[C96] D. Coppersmith. *Finding a Small Root of a Univariate Modular Equation*. Proceedings of EUROCRYPT ’96, LNCS 1070, pp. 155-165, Springer-Verlag (1996).

[FR95] M. Franklin and M. Reiter. *A Linear Protocol Failure for RSA with Exponent Three*. Présenté à la rump session, Crypto ’95. (Source citée par Coppersmith).

[HG97] N. Howgrave-Graham. *Finding Small Roots of Univariate Modular Equations Revisited*. Cryptography and Coding, LNCS 1355, pp. 131-142, Springer-Verlag (1997).

[May09] A. May. *Using LLL-reduction for solving RSA and factorization problems*. Chapitre 11 de The LLL Algorithm - Survey and Applications, pp. 315-348, Springer (2009).